

UI 프론트엔드 프로그래밍 과정

jQuery

jQuery 기본

1 개요

◆ jQuery

- 모든 브라우저에서 동작하는 클라이언트 자바스크립트 라이브러리
- 2006년 1월, 존 레식 John Resig이 BarCamp NYC에서 발표
- 무료로 사용 가능한 오픈 소스 라이브러리
- jQuery는 다음 기능을 위해 제작됨
 - 문서 객체 모델과 관련된 처리를 쉽게 구현 -> :
 - 일관된 이벤트 연결을 쉽게 구현
 - 시각적 효과를 쉽게 구현
 - Ajax 애플리케이션을 쉽게 개발

2 다운로드

◆ jQuery 다운로드

- jQuery를 내려받으려면 <http://jquery.com>에 접속
- 메인 화면에서 다운로드 버튼을 누르면 다운로드 페이지로 이동
- 다운로드 페이지에 jQuery 1.X와 jQuery 2.X의 2가지 버전 중 1.X 버전으로 내려받음

2 다운로드

◆ jQuery 사용

- 첫 번째 방법은 CDN 호스트를 사용하는 방법
- 두 번째는 직접 내려받아 사용하는 방법

Using jQuery with a CDN

CDNs can offer a performance benefit by hosting jQuery on servers spread across the globe. This also offers an advantage that if the visitor to your webpage has already downloaded a copy of jQuery from the same CDN, it won't have to be re-downloaded.

jQuery's CDN provided by MediaTemple

To use the jQuery CDN, just reference the file directly from `http://code.jquery.com` in the script tag:

```
1 | <script src="http://code.jquery.com/jquery-1.10.1.min.js"></script>
2 | <script src="http://code.jquery.com/jquery-migrate-1.2.1.min.js"></script>
```

- 구글과 마이크로소프트에서 CDN 호스트를 지원
- CDN : Content Delivery Network은 사용자에게 간편하게 콘텐츠를 제공하는 방식

2 다운로드

◆ jQuery 사용

- 첫 번째 방법은 CDN 호스트를 사용하는 방법
 - 구글과 마이크로소프트에서 CDN 호스트를 지원
- 두 번째는 직접 내려받아 사용하는 방법

Using jQuery with a CDN

CDNs can offer a performance benefit by hosting jQuery on servers spread across the globe. This also offers an advantage that if the visitor to your webpage has already downloaded a copy of jQuery from the same CDN, it won't have to be re-downloaded.

jQuery's CDN provided by MediaTemple

To use the jQuery CDN, just reference the file directly from `http://code.jquery.com` in the script tag:

```
1 | <script src="http://code.jquery.com/jquery-1.10.1.min.js"></script>
2 | <script src="http://code.jquery.com/jquery-migrate-1.2.1.min.js"></script>
```

2 다운로드

◆ CDN 호스트 사용

```
<head>
  <script src="http://code.jquery.com/jquery-1.10.2.js"></script>
  <script>

  </script>
</head>
```

■ 그 밖의 CDN 호스트

- <http://code.jquery.com/jquery-1.10.2.js>
- <http://code.jquery.com/jquery-1.10.2.min.js>
- <https://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.js>
- <https://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js>
- <http://ajax.aspnetcdn.com/ajax/jQuery/jquery-1.10.2.js>
- <http://ajax.aspnetcdn.com/ajax/jquery/jquery-1.10.2.min.js>

2 다운로드

◆ CDN 호스트 사용

- 하이브리드 애플리케이션 같은 오프라인 환경에서 jQuery를 사용한다면 반드시 내려받아 사용

```
<!DOCTYPE html>
<html>
<head>
  <script src="jquery-1.10.2.js"></script>
  <script>

    </script>
</head>
<body>

</body>
</html>
```

3 \$(document).ready()

◆ \$(document).ready()

- jQuery를 사용한 모든 웹 페이지는 다음 코드로 시작

```
<script>                                (      )  
    $(document).ready(function () {  
        jQuery                ready    onload  
    });  
</script>
```

3 \$(document).ready()

◆ \$(document).ready()

- 이벤트 연결

```
<script>
    window.onload = function () {

    };
</script>
```

```
<script>
    $(document).ready(function () {
        alert('First READY');
    });

    $(document).ready(function () {
        alert('Second READY');
    });

    $(document).ready(function () {
        alert('Third READY');
    });
</script>
```

4 기본 선택자

◆ 기본 선택자

- jQuery 메서드의 가장 기본적인 형태
- 선택자는 jQuery에서 가장 중요한 역할

`$('hl').css('color', 'red');`

\$ 선택자 메시지

4 기본 선택자

◆ 전체 선택자

- CSS의 가장 기본적인 선택자는 전체 선택자
- *를 전체 선택자라고 부름

```
<!DOCTYPE html>
<html>
<head>
  <script src="http://code.jquery.com/jquery-1.10.2.js"></script>
  <script>
    $(document).ready(function () {
      $('*').css('color', 'red');
    });
  </script>
</head>
<body>
  <h1>Lorem ipsum</h1>
</body>
</html>
```

4 기본 선택자

◆ 태그 선택자

- 특정한 태그 선택하는 선택자
- 태그 선택자는 태그의 이름을 그냥 사용

```
▼<body>
  <h1 style="color: orange; ">Lorem ipsum</h1>
  <p>Lorem ipsum dolor sit amet.</p>
  <h1 style="color: orange; ">Lorem ipsum</h1>
  <p>consectetur adipiscing elit.</p>
</body>
```

4 기본 선택자

◆ 아이디 선택자

- 아이디 선택자는 특정한 id 속성이 있는 문서 객체를 선택하는 선택자
- 두 번째에 위치한 h1 태그가 id 속성으로 target을 가짐

```
<script>
  $(document).ready(function () {
    $('#target').css('color', 'orange');
  });
</script>
```

```
<script>
  $(document).ready(function () {
    $('h1#target').css('color', 'orange');
  });
</script>
```

4 기본 선택자

◆ 클래스 선택자

- 아클래스 선택자는 특정한 class 속성이 있는 문서 객체를 선택하는 선택자

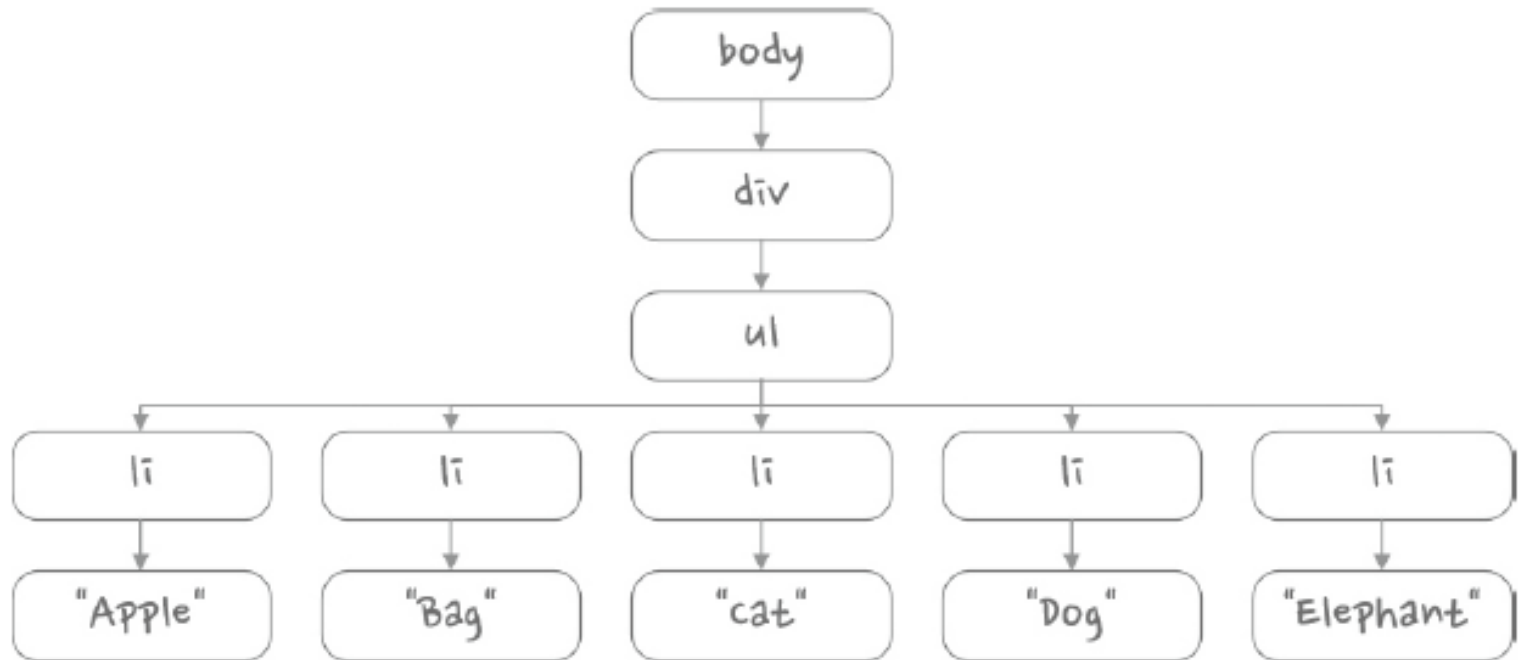
```
<script>
$(document).ready(function () {
    $('.item').css('color', 'orange');
    $('h1.item').css('background', 'red');
});
</script>
```

```
<script>
$(document).ready(function () {
    $('.item.select').css('color', 'orange');
});
</script>
```

5 자손 선택자와 후손 선택자

◆ 자손 선택자와 후손 선택자

- 자손 선택자와 후손 선택자는 기본 선택자의 앞에 붙여 사용하며 기본 선택자의 범위를 제한



5 자손 선택자와 후손 선택자

◆ 자손 선택자

- 자손 선택자는 자손을 선택하는 선택자이며 '요소A > 요소B'의 형태로 사용

```
<script>
  $(document).ready(function () {
    $('body > *').css('color', 'red');
  });
</script>
```

5 자손 선택자와 후손 선택자

◆ 후손 선택자

- 후손 선택자는 이름 그대로 후손을 선택하는 선택자
- '요소A 요소B'의 형태로 사용하며 요소A의 후손으로 범위를 한정

```
▼ <html>
  ▶ <head>...</head>
  ▼ <body>
    ▼ <div style="color: red; ">
      ▼ <ul style="color: red; ">
        <li style="color: red; ">Apple</li>
        <li style="color: red; ">Bag</li>
        <li style="color: red; ">Cat</li>
        <li style="color: red; ">Dog</li>
        <li style="color: red; ">Elephant</li>
      </ul>
    </div>
  </body>
</html>
```

6 속성 선택자

◆ 후손 선택자

- 속성 선택자는 기본 선택자 뒤에 붙여 사용

선택자 형태	설명
요소[속성=값]	속성과 값이 같은 문서 객체를 선택합니다.
요소[속성 =값]	속성 안의 값이 특정 값과 같은 문서 객체를 선택합니다.
요소[속성~=값]	속성 안의 값이 특정 값을 단어로 시작하는 문서 객체를 선택합니다.
요소[속성^=값]	속성 안의 값이 특정 값으로 시작하는 문서 객체를 선택합니다.
요소[속성\$=값]	속성 안의 값이 특정 값으로 끝나는 문서 객체를 선택합니다.
요소[속성*=값]	속성 안의 값이 특정 값을 포함하는 문서 객체를 선택합니다.

6 속성 선택자

◆ 후손 선택자

- 속성 선택자는 기본 선택자 뒤에 붙여 사용

```
<script>
  $(document).ready(function () {
    $('input[type="text"]').val('Hello jQuery..!');
  });
</script>
```

7 필터 선택자

◆ 입력 양식 필터 선택자

- 필터 선택자는 기본 선택자 뒤에 사용

선택자 형태	설명
요소:button	input 태그 중 type 속성이 button인 문서 객체와 button 태그를 선택합니다.
요소:checkbox	input 태그 중 type 속성이 check인 문서 객체를 선택합니다.
요소:file	input 태그 중 type 속성이 file인 문서 객체를 선택합니다.
요소:image	input 태그 중 type 속성이 image인 문서 객체를 선택합니다.
요소:password	input 태그 중 type 속성이 password인 문서 객체를 선택합니다.
요소:radio	input 태그 중 type 속성이 radio인 문서 객체를 선택합니다.
요소:reset	input 태그 중 type 속성이 reset인 문서 객체를 선택합니다.
요소:submit	input 태그 중 type 속성이 submit인 문서 객체를 선택합니다.
요소:text	input 태그 중 type 속성이 text인 문서 객체를 선택합니다.

7 필터 선택자

◆ 입력 양식 필터 선택자

선택자 형태	설명
요소:checked	체크되어 있는 입력 양식을 선택합니다.
요소:disabled	비활성화된 입력 양식을 선택합니다.
요소:enabled	활성화된 입력 양식을 선택합니다.
요소:focus	초점이 맞추어져 있는 입력 양식을 선택합니다.
요소:input	모든 입력 양식을 선택합니다(input, textarea, select, button 태그).
요소:selected	option 객체 중 선택된 태그를 선택합니다.

```
<script>
$(document).ready(function () {
    // 5초 후에 코드를 실행합니다.
    setTimeout(function () {
        // 변수를 선언합니다.
        var value = $('select > option:selected').val();

        // 출력합니다.
        alert(value);
    }, 5000);
});
</script>
```

7 필터 선택자

◆ 위치 필터 선택자

선택자 형태	설명
요소:odd	홀수 번째에 위치한 문서 객체를 선택합니다.
요소:even	짝수 번째에 위치한 문서 객체를 선택합니다.
요소:first	첫 번째에 위치한 문서 객체를 선택합니다.
요소:last	마지막에 위치한 문서 객체를 선택합니다.

```
<script>
    $(document).ready(function () {
        $('tr:odd').css('background', '#F9F9F9');
        $('tr:even').css('background', '#9F9F9F');

        $('tr:first').css('background', '#000000').css('color', #FFFFFF);
    });
</script>
```

8 배열 관리

◆ 배열 관리

- jQuery로 배열을 관리할 때는 `each ()` 메서드를 사용
- `each ()` 메서드는 매개변수로 입력한 함수로 `for in` 반복문처럼 객체나 배열의 요소를 검사하는 메서드
- `each ()` 메서드는 다음과 같이 두 가지 형태로 사용

1 `$each(object, function(index, item) { })`

2 `$(selector).each(function(index, item) { })`

8 배열 관리

◆ 자바스크립트 배열 관리

- \$.each() 메서드
 - \$.each () 메서드의 첫 번째 매개변수에는 배열을 넣음
 - 두 번째 매개변수는 매개변수로 index와 item을 갖는 함수를 넣음

```
<script>
    $(document).ready(function () {
        // 변수를 선언합니다.
        var array = [
            { name: 'Hanbit Media', link: 'http://hanb.co.kr' },
            { name: 'Naver', link: 'http://naver.com' },
            { name: 'Daum', link: 'http://daum.net' },
            { name: 'Paran', link: 'http://paran.com' }
        ];

        // $.each() 메서드를 사용합니다.
        $.each(array, function (index, item) {
        });
    });
</script>
```

8 배열 관리

◆ 자바스크립트 배열 관리

- \$.each() 메서드의 콜백 함수

```
// $.each() 메서드를 사용합니다.
$.each(array, function (index, item) {
    // 변수를 선언합니다.
    var output = '';

    // 문자열을 만듭니다.
    output += '<a href="' + item.link + '">';
    output += '    <h1>' + item.name + '</h1>';
    output += '</a>';

    // 집어넣습니다.
    document.body.innerHTML += output;
});
```

8 배열 관리

◆ jQuery 배열 관리

- jQuery의 배열 객체는 따로 만드는 것이 아니라, 선택자로 여러 개의 문서 객체를 선택할 때 생성

```
<body>
  <h1>item - 0</h1>
  <h1>item - 1</h1>
  <h1>item - 2</h1>
  <h1>item - 3</h1>
  <h1>item - 4</h1>
</body>
</html>
```

8 배열 관리

◆ jQuery 배열 관리

- `addClass()` 메서드

```
<script>
    $(document).ready(function () {
        $('h1').addClass('high-light');
    });
</script>
```

8 배열 관리

◆ jQuery 배열 관리

- \$(selector).each() 메서드

```
<script>
    $(document).ready(function () {
        $('h1').each(function (index, item) {

            });
    });
</script>
```

문서 객체 선택과 탐색

1 기본 필터 메서드

◆ 기본 필터 메서드

- 기본 필터 메서드

메서드 이름	설명
filter()	문서 객체를 필터링합니다.

- 두 가지 형태

1 `$(selector).filter(selector);`

2 `$(selector).filter(function () { });`



return boolean

1 기본 필터 메서드

◆ 기본 필터 메서드

- 필터 선택자

```
<script>
    $(document).ready(function () {
        $('h3:even').css({
            backgroundColor: 'black',
            color: 'white'
        });
    });
</script>
```

```
<script>
    $(document).ready(function () {
        $('h3').filter(':even').css({
            backgroundColor: 'black',
            color: 'white'
        });
    });
</script>
```

1 기본 필터 메서드

◆ 기본 필터 메서드

- filter () 메서드의 2번 형태로 매개변수 index가 3의 배수인 h3 태그를 선택

```
<script>
    $(document).ready(function () {
        $('h3').filter(function (index) {
            return index % 3 == 0;
        }).css({
            backgroundColor: 'black',
            color: 'white'
        });
    });
</script>
```

2 문서 객체 탐색 종료

◆ 문서 객체 탐색 종료

- 체이닝을 사용할 때 추가한 filter() 메서드를 제거하려면 end() 메서드를 사용

메서드 이름	설명
end()	문서 객체 선택을 한 단계 뒤로 돌립니다.

```
$('#h1').css('background', 'orange').filter(':even').css('color', 'white').end().  
filter(':odd').css('color', 'red');
```

3 특정 위치의 문서 객체 선택

◆ 특정 위치의 문서 객체 선택

- 필터 선택자를 이용하면 특정 위치에 존재하는 문서 객체를 선택할 수 있음

메서드 이름	설명
eq()	특정 위치에 존재하는 문서 객체를 선택합니다.
first()	첫 번째에 위치하는 문서 객체를 선택합니다.
last()	마지막에 위치하는 문서 객체를 선택합니다.

```
<script>
    $(document).ready(function () {
        $('h1').eq(0).css('background', 'orange');
        $('h1').eq(-1).css('background', 'red');
    });
</script>
```

4 문서 객체 추가 선택

◆ 문서 객체 추가 선택

- jQuery는 문서 객체의 체이닝을 더 유연하게 하려고 add () 메서드 제공
- add () 메서드를 사용하면 현재 선택한 문서 객체의 범위를 확장할 수 있음

메서드 이름	설명
add()	문서 객체를 추가적으로 선택합니다.

```
<script>
    $(document).ready(function () {
        $('h1').css('background', 'Gray').add('h2').css('float','left');
    });
</script>
```

5 특정 태그 선택

◆ 특정 태그 선택

- XML 문자열 생성

```
<script>
    // 변수를 선언합니다.
    var xml = '';
    xml += '<friends>';
    xml += '    <friend>';
    xml += '        <name>연하진</name>';
    xml += '        <language>Ruby</language>';
    xml += '    </friend>';
    xml += '    <friend>';
    xml += '        <name>윤명월</name>';
    xml += '        <language>Basic</language>';
    xml += '    </friend>';
    xml += '    <friend>';
    xml += '        <name>윤하린</name>';
    xml += '        <language>C#</language>';
    xml += '    </friend>';
    xml += '</friends>';
```

5 특정 태그 선택

◆ 특정 태그 선택

- **\$.parseXML()** 메서드와 **each()** 메서드, **find()** 메서드

```
$(document).ready(function () {  
    // 변수를 선언합니다.  
    var xmlDoc = $.parseXML(xml);  
    $(xmlDoc).find('friend').each(function (index) {  
  
        });  
    });  
});
```

5 특정 태그 선택

◆ 특정 태그 선택

- XML 파싱

```
var xmlDoc = $.parseXML(xml);
$(xmlDoc).find('friend').each(function (index) {
    // 변수를 선언합니다.
    var output = '';
    output += '<div>';
    output += '    <h1>' + $(this).find('name').text() + '</h1>';
    output += '    <p>' + $(this).find('language').text() + '</p>';
    output += '</div>';

    // 출력합니다.
    document.body.innerHTML += output;
});
});
```

문서 객체 조작

1 문서 객체의 클래스 속성 추가

◆ 문서 객체의 클래스 속성 추가

- 문서 객체에 클래스 속성을 추가할 때 사용하는 메서드

메서드 이름	설명
addClass()	문서 객체의 클래스 속성을 추가합니다.

```
<script>
    $(document).ready(function () {
        $('h1').addClass('item');
    });
</script>
```

2 문서 객체의 클래스 속성 제거

◆ 문서 객체의 클래스 속성 제거

- 문서 객체에 클래스 속성을 제거할 때 사용하는 메서드

메서드 이름	설명
removeClass()	문서 객체의 클래스 속성을 제거합니다.

```
<script>
  $(document).ready(function () {
    $('h1').removeClass('select');
  });
</script>
```

3 문서 객체의 클래스 속성 검사

◆ 문서 객체의 클래스 속성 검사

- Query에서 문서 객체의 속성과 관련된 모든 기능은 attr() 메서드가 처리

메서드 이름	설명
attr()	속성과 관련된 모든 기능을 수행합니다.

```
<script>
$(document).ready(function () {
    // 변수를 선언합니다.
    var src = $('img').attr('src');

    // 출력합니다.
    alert(src);
});
</script>
```

4 문서 객체의 속성 추가

◆ 문서 객체의 속성 추가

- 문서 객체에 속성을 추가할 때도 attr () 메서드를 사용
- attr () 메서드는 다음과 같은 세 가지 형태로 사용

1 \$(selector).attr(name, value);

2 \$(selector).attr(name, function(index, attr) { });

3 \$(selector).attr(object);

4 문서 객체의 속성 추가

◆ 문서 객체의 속성 추가

- attr() 메서드 - Setter(1)

```
<script>
    $(document).ready(function () {
        $('img').attr('width', 200);
    });
</script>
```

4 문서 객체의 속성 추가

◆ 문서 객체의 속성 추가

- attr() 메서드 - Setter(2)

```
<script>
    $(document).ready(function () {
        $('img').attr('width', function (index) {
            return (index + 1) * 100;
        });
    });
</script>
```

4 문서 객체의 속성 추가

◆ 문서 객체의 속성 추가

- attr() 메서드 - Setter(3)

```
<script>
    $(document).ready(function () {
        $('img').attr({
            width: function (index) {
                return (index + 1) * 100;
            },
            height: 100
        });
    });
</script>
```

5 문서 객체의 속성 제거

◆ 문서 객체의 속성 제거

- 문서 객체의 속성을 제거할 때는 아래의 메서드를 사용

메서드 이름	설명
removeAttr(name)	문서 객체의 속성을 제거합니다.

```
<script>
    $(document).ready(function () {
        $('h1').removeAttr('data-index');
    });
</script>
```


6 문서 객체의 스타일 검사

◆ 문서 객체의 스타일 검사

메서드 이름	설명
css()	스타일과 관련된 모든 기능을 수행합니다.

```
<script>
    $(document).ready(function () {
        // 변수를 선언합니다.
        var color = $('h1').css('color');

        // 출력합니다.
        alert(color);
    });
</script>
```

6 문서 객체의 스타일 추가

◆ 문서 객체의 스타일 추가

- 지금까지 사용한 형태외에도 다음 형태로 css () 메서드를 사용할 수 있음

1 `$(selector).css(name, value);`

2 `$(selector).css(name, function(index, style) { });`

3 `$(selector).css(object);`

```
<script>
    $(document).ready(function () {
        $('h1').css('color', 'red');
    });
</script>
```

6 문서 객체의 스타일 추가

◆ 문서 객체의 스타일 추가

```
<script>
    $(document).ready(function () {
        // 변수를 선언합니다.
        var color = ['red', 'white', 'purple'];

        // 문서 객체의 스타일을 변경합니다.
        $('h1').css('color', function (index) {
            return color[index];
        });
    });
</script>
```

6 문서 객체의 스타일 추가

◆ 문서 객체의 스타일 추가

```
<script>
  $(document).ready(function () {
    // 변수를 선언합니다.
    var color = ['red', 'white', 'purple'];

    // 문서 객체의 스타일을 변경합니다.
    $('h1').css({
      color: function (index) {
        return color[index];
      },
      backgroundColor: 'black'
    });
  });
</script>
```

7 문서 객체의 내부 검사

◆ 문서 객체의 내부 검사

- 기존의 자바스크립트에서 문서 객체의 innerHTML, textContent 속성과 관련된 jQuery 메서드가 표 15-6의 메서드

메서드 이름	설명
html()	문서 객체 내부의 글자와 관련된 모든 기능을 수행합니다(HTML 태그 인식).
text()	문서 객체 내부의 글자와 관련된 모든 기능을 수행합니다.

7 문서 객체의 내부 검사

◆ 문서 객체의 내부 검사

- html() 메서드 - Getter

```
<script>
    $(document).ready(function () {
        // 변수를 선언합니다.
        var html = $('h1').html();

        // 출력합니다.
        alert(html);
    });
</script>
```

8 문서 객체의 내부 검사

◆ 문서 객체의 내부 검사

- text() 메서드 - Getter

```
<script>
    $(document).ready(function () {
        // 변수를 선언합니다.
        var text = $('h1').text();

        // 출력합니다.
        alert(text);
    });
</script>
```

9 문서 객체의 내부 추가

◆ 문서 객체의 내부 추가

- 문서 객체의 내부에 내용물을 추가하고 싶을 때도 `html ()` 메서드와 `text ()` 메서드를 사용
- 두 메서드 모두 다음과 같은 형태로 사용

```
1 $(selector).html(value);
```

```
   $(selector).text(value);
```

```
2 $(selector).html(function(index, html) { });
```

```
   $(selector).text(function(index, text) { });
```


9 문서 객체의 내부 추가

◆ 문서 객체의 내부 추가

```
<script>
    $(document).ready(function () {
        $('div').html('<h1>$.html() Method</h1>');
    });
</script>
```

```
<script>
    $(document).ready(function () {
        $('div').html(function (index) {
            return '<h1>Header-' + index + '</h1>';
        });
    });
</script>
```

10 문서 객체 제거

◆ 문서 객체 제거

- 문서 객체를 제거할 때는 아래의 메서드를 사용

메서드 이름	설명
remove()	문서 객체를 제거합니다.
empty()	문서 객체 내부를 비웁니다.

```
<script>
    $(document).ready(function () {
        $('div').empty();
    });
</script>
```

11 문서 객체 생성(1)

◆ 문서 객체 생성(1)

- 문서 객체를 생성할 때는 아래의 메서드를 사용
- jQuery () 메서드는 선택자로 문서 객체를 선택하는 기능 이외에도 문서 객체를 생성하는 기능이 있음

메서드 이름	설명
<code>\$()</code>	문서 객체를 생성합니다.

11 문서 객체 생성(1)

◆ 문서 객체 생성(1)

- \$ () 메서드의 매개변수에 HTML 태그를 문자열로 넣기만 하면 문서 객체가 생성 과 문서 객체 생성 및 텍스트 노드 추가

```
<script>
  $(document).ready(function () {
    $('<h1></h1>');
  });
</script>
```

```
<script>
  $(document).ready(function () {
    $('<h1></h1>').html('Hello World .. !');
  });
</script>
```

11 문서 객체 생성(1)

◆ 문서 객체 생성(1)

- 문서 객체 생성 , 문서 객체 연결

```
<script>
  $(document).ready(function () {
    $('<h1></h1>').html('Hello World .. !').appendTo('body');
  });
</script>
```

```
<script>
  $(document).ready(function () {
    $('<h1>Hello World .. !</h1>').appendTo('body');
  });
</script>
```

12 문서 객체 생성(2)

◆ 문서 객체 생성(2)

- 텍스트 노드를 갖지 않는 문서 객체를 생성하는 방법
- img 태그를 생성할 때는 \$ () 메서드로 문서 객체를 생성하고 attr () 메서드로 속성을 입력

```
<script>
    $(document).ready(function () {
        $('<img />').attr('src', 'Chrysanthemum.jpg').appendTo('body');
    });
</script>
```

12 문서 객체 생성(2)

◆ 문서 객체 생성(2)

```
<script>
  $(document).ready(function () {
    $('<img />', {
      src: 'Chrysanthemum.jpg',
      width: 350,
      height: 250
    }).appendTo('body');
  });
</script>
```

13 문서 객체 삽입(1)

◆ 문서 객체 삽입(1)

- jQuery에는 문서 객체에 문서 객체를 추가하는 메서드가 여덟 개
- 여덟 개의 메서드는 크게 두 가지 형태로 나눌 수 있음

메서드 이름	설명
\$(A).appendTo(B)	A를 B의 뒷부분에 추가합니다.
\$(A).prependTo(B)	A를 B의 앞부분에 추가합니다.
\$(A).insertAfter(B)	A를 B의 뒤에 추가합니다.
\$(A).insertBefore(B)	A를 B의 앞에 추가합니다.



14 문서 객체 삽입(2)

◆ 문서 객체 삽입(2)

- 문서 객체 삽입(1)과 반대의 순서로 문서 객체를 추가하는 메서드

메서드 이름	설명
<code>\$(A).append(B)</code>	B를 A의 뒷부분에 추가합니다.
<code>\$(A).prepend(B)</code>	B를 A의 앞부분에 추가합니다.
<code>\$(A).after(B)</code>	B를 A의 뒤에 추가합니다.
<code>\$(A).before(B)</code>	B를 A의 앞에 추가합니다.

- `append ()` 메서드의 사용 형태

가 가

1 `$(selector).append(content, content, , content)`

2 `$(selector).append(function (index) { });`

14 문서 객체 삽입(2)

◆ 문서 객체 삽입(2)

- 첫 번째 형태는 여러 개의 문서 객체를 한꺼번에 입력할 수 있음

```
<script>
    $(document).ready(function () {
        // 변수를 선언합니다.
        var h1 = '<h1>Header1</h1>';
        var h2 = '<h2>Header2</h2>';

        // 문서 객체를 추가합니다.
        $('body').append(h1, h2, h1, h2);
    });
</script>
```

14 문서 객체 삽입(2)

◆ 문서 객체 삽입(2)

- 두 번째 형태는 append () 메서드의 매개변수에 index 매개변수를 갖는 함수를 넣어줌
- 각각의 div 태그에 다른 내용을 쉽게 입력하려고 배열을 선언

```
<script>
    $(document).ready(function () {
        // 변수를 선언합니다.
        var content = [
            { name: '윤인성', region: '서울특별시 강서구' },
            { name: '윤하린', region: '서울특별시 광진구' },
            { name: '윤인아', region: '미국 메사추세츠' }
        ];
```

14 문서 객체 삽입(2)

◆ 문서 객체 삽입(2)

- 배열을 사용한 문서 객체 생성과 추가

```
// 문서 객체를 추가합니다.  
$('div').append(function (index) {  
    // 변수를 선언합니다.  
    var item = content[index];  
    var output = '';  
  
    output += '<h1>' + item.name + '</h1>';  
    output += '<h2>' + item.region + '</h2>';  
  
    return output;  
});
```

```
▼ <body>  
  ▼ <div>  
    <h1>윤인성</h1>  
    <h2>서울특별시 강서구</h2>  
  </div>  
  ▼ <div>  
    <h1>윤하린</h1>  
    <h2>서울특별시 광진구</h2>  
  </div>  
  ▼ <div>  
    <h1>윤민마</h1>  
    <h2>미국 메사추세츠</h2>  
  </div>  
</body>
```

15 문서 객체 이동

◆ 문서 객체 이동

- 기존 문서 객체를 선택하고 문서 객체 삽입 메서드를 사용하면 문서 객체를 쉽게 다른 곳으로 이동시킬 수 있음
- 시간에 따라 이미지의 순서를 지속적으로 변경하는 간단한 예제

```
<body>
  
  
  
  
  
</body>
```

15 문서 객체 이동

◆ 문서 객체 이동

- appendTo() 메서드를 사용한 문서 객체의 이동
- 지속적인 이미지 순서 변경

```
<script>
    $(document).ready(function () {
        $('img').first().appendTo('body');
    });
</script>
```

```
<script>
    $(document).ready(function () {
        // .image의 크기를 조정합니다.
        $('img').css('width', 250);

        // 함수를 2초마다 실행합니다.
        setInterval(function () {
            // 첫 번째 이미지를 마지막으로 보냅니다.
            $('img').first().appendTo('body');
        }, 2000);
    });
</script>
```

이벤트

이벤트

◆ 이벤트

- jQuery의 이벤트에는 기존 자바스크립트의 이벤트가 모두 존재
- jQuery를 사용하면 기존 자바스크립트의 이벤트를 연결할 때보다 훨씬 간편하게 이벤트를 연결할 수 있음

```
$(document).ready(function (event) {  
  
});
```

1 이벤트 연결 기본

◆ 이벤트 연결 기본

- jQuery로 이벤트를 연결하는 가장 기본적인 방법은 on () 메서드 사용

메서드 이름	설명
on ()	이벤트를 연결합니다.

- on () 메서드 사용 형태

1 `$(selector).on(ex) clickeventName, handlerfunction(event) { })`

2 `$(selector).on(object)`

1 이벤트 연결 기본

◆ 이벤트 연결 기본

- on () 메서드
- h1 태그를 click 이벤트에 연결하고 이벤트 발생 시 이벤트 발생 객체에 '+' 글자 추가

가
가

```
<script>
    $(document).ready(function () {
        // 이벤트를 연결합니다.
        $('h1').on('click', function () {
            $(this).html(function (index, html) {
                return html + '+';
            });
        });
    });
</script>
```

1 이벤트 연결 기본

◆ 이벤트 연결 기본

- on () 메서드
- on () 메서드의 매개변수에 객체를 넣어줌
- 속성 이름과 속성 값에 이벤트 이름과 이벤트 리스너를 넣으면 이벤트를 쉽게 연결

```
// 이벤트를 연결합니다.  
$('h1').on({  
    mouseenter: function () { $(this).addClass('reverse'); },  
    mouseleave: function () { $(this).removeClass('reverse'); }  
});  
});  
</script>
```

2 간단한 이벤트 연결

◆ 간단한 이벤트 연결

blur	focus	focusin	focusout	load
resize	scroll	unload	click	dblclick
mousedown	mouseup	mousemove	mouseover	mouseout
mouseenter	mouseleave	change	select	submit
keydown	keypress	keyup	error	ready

- 간단한 방식으로 이벤트를 연결할 때는 다음 방법을 사용

```
$(selector).method(function(event) { });
```

2 간단한 이벤트 연결

◆ 간단한 이벤트 연결

- jQuery는 이벤트 연결 메서드도 제공

메서드 이름	설명
<code>hover()</code>	<u>mouseenter 이벤트</u> 와 <u>mouseleave 이벤트</u> 를 동시에 연결합니다.

```
$(selector).hover(function(event) { }, function(event) { });
```

2 간단한 이벤트 연결

◆ 간단한 이벤트 연결

- hover() 메서드

```
<script>
    $(document).ready(function () {
        // 이벤트를 연결합니다.
        $('h1').hover(function () {
            $(this).addClass('reverse');
        }, function () {
            $(this).removeClass('reverse');
        });
    });
</script>
```

3 이벤트 연결 제거

◆ 이벤트 연결 제거

- 이벤트를 제거할 때는 off () 메서드를 사용

메서드 이름	설명
off ()	이벤트를 제거합니다.

- off () 메서드는 다음 형태로 사용
- 1번 형태는 해당 문서 객체와 관련된 모든 이벤트를 제거
- 2번 형태는 해당 문서 객체의 특정 이벤트와 관련된 모든 이벤트를 제거
- 3번 형태는 특정 이벤트 리스너를 제거

1 `$(selector).off()`

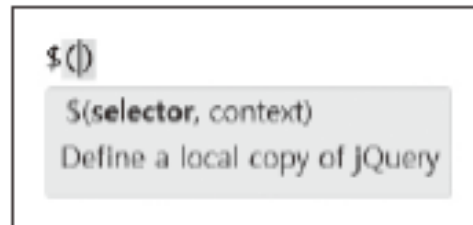
2 `$(selector).off(eventName)`

3 `$(selector).off(eventName, function)`

4 매개변수 context

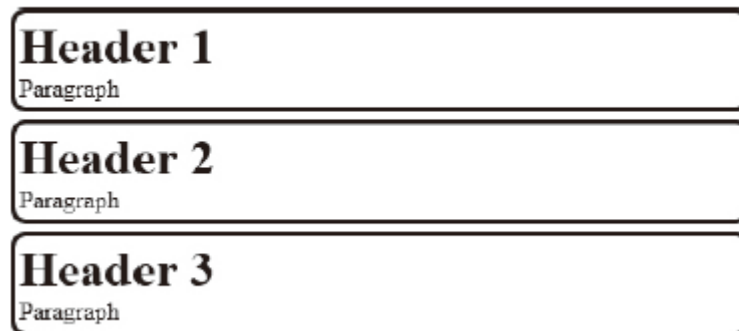
◆ 매개변수 context

- jQuery 메서드는 사실 매개변수를 두 개 입력할 수도 있음
- 특정 부분에 선택자를 적용하고 싶을 때 사용하는 것이 매개변수 context



- 매개변수 context는 selector가 적용하는 범위를 한정

그림 16-5 문제



4 매개변수 context

◆ 매개변수 context

- context 객체 : **this**

```
<script>
  $(document).ready(function () {
    // 이벤트를 연결합니다.
    $('div').click(function () {
      // 변수를 선언합니다.
      var header = $('h1', this).text();
      var paragraph = $('p', this).text();

      // 출력합니다.
      alert(header + '\n' + paragraph);
    });
  });
</script>
```

5 이벤트 객체

가 /

◆ 이벤트 객체 `function(e){ }`

- 모든 이벤트 리스너는 이벤트 객체가 있음
- jQuery가 스스로 이벤트 객체를 정형화하므로 jQuery의 이벤트 객체는 모든 브라우저가 같은 방법으로 사용하고 같은 속성을 가짐
- 자주 사용하는 jQuery 이벤트 객체

이벤트 객체 속성	설명
<code>event.pageX</code>	브라우저의 화면을 기준으로 한 마우스의 X 좌표 위치
<code>event.pageY</code>	브라우저의 화면을 기준으로 한 마우스의 Y 좌표 위치
<code>event.preventDefault()</code>	기본 이벤트를 제거합니다. <code>return false</code> 가
<code>event.stopPropagation()</code>	이벤트 전달을 제거합니다.

6 이벤트 강제 발생

`.onclick();`

◆ 이벤트 강제 발생

- 이벤트 강제 발생 메서드

메서드 이름	설명
<code>trigger()</code>	이벤트를 강제로 발생시킵니다.

- `trigger ()` 메서드는 다음과 같은 형태로 사용
- 2번 형태의 매개변수 `data`는 일반적으로 배열을 집어넣음

1 `$(selector).trigger(eventName)`

2 `$(selector).trigger(eventName, data)`

6 이벤트 강제 발생

◆ 이벤트 강제 발생

- trigger() 메서드

```
<script>
    $(document).ready(function () {
        // 이벤트를 연결합니다.
        $('#h1').click(function () {
            $(this).html(function (index, html) {
                return html + '★';
            });
        });

        // 1초마다 함수를 실행합니다.
        setInterval(function () {
            $('#h1').last().trigger('click');
        }, 1000);
    });
</script>
```

7 기본 이벤트와 이벤트 전달

◆ 기본 이벤트와 이벤트 전달

- **기본 이벤트를 제거하고 이벤트 전달을 막을 때**는 이벤트 객체에 있는 아래의 메서드를 사용

메서드 이름	설명
<code>preventDefault()</code>	기본 이벤트를 제거합니다.
<code>stopPropagation()</code>	이벤트 전달을 제거합니다.

7 기본 이벤트와 이벤트 전달

◆ 기본 이벤트와 이벤트 전달

- preventDefault () 메서드로 a 태그의 기본 이벤트를 제거

```
<script>
$(document).ready(function () {
    $('a').click(function (event) {
        $(this).css('background-color', 'blue');
        event.preventDefault();
    });

    $('h1').click(function () {
        $(this).css('background-color', 'red');
    });
});
</script>
```



7 기본 이벤트와 이벤트 전달

◆ 기본 이벤트와 이벤트 전달

- 이벤트 전달을 막으려면 이벤트 객체의 stopPropagation () 메서드를 사용

```
$('#a').click(function (event) {  
    $(this).css('background-color', 'blue');  
    event.stopPropagation();  
    event.preventDefault();  
});
```



7 기본 이벤트와 이벤트 전달

◆ 기본 이벤트와 이벤트 전달

- `return false` : `stopPropagation ()` 메서드와 `preventDefault ()` 메서드를 함께 사용하는 경우가 많으므로, jQuery는 간단하게 `return false`를 사용하면 이 두 가지를 함께 적용하는 것으로 인식

```
$('#a').click(function (event) {  
    $(this).css('background-color', 'blue');  
    return false;  
});
```

8 이벤트 연결 범위 한정

◆ 이벤트 연결 범위 한정

- 이벤트를 연결할 때 on () 메서드를 사용

```
$( 'h1' ).on()  
on(types, selector, data, fn, one)
```

- 기본 이벤트 연결

```
<script>  
  $(document).ready(function () {  
    $('h1').on('click', function () {  
      var length = $('h1').length;  
      var targetHTML = $(this).html();  
      $('#wrap').append('<h1>' + length + ' - ' + targetHTML + '</h1>');  
    });  
  });  
</script>
```

8 이벤트 연결 범위 한정

◆ 이벤트 연결 범위 한정

- 맨 위의 h1 태그를 클릭하면 요소가 추가됨

Header

1 - Header

2 - Header

3 - Header

8 이벤트 연결 범위 한정

◆ 이벤트 연결 범위 한정

- delegate 방식을 사용하는 on() 메서드
- 상위 태그에 이벤트를 연결하고 "h1 태그를 클릭했을 때"를 검출

```
<script>
    $(document).ready(function () {
        $('#wrap').on('click', 'h1', function () {
            var length = $('h1').length;
            var targetHTML = $(this).html();
            $('#wrap').append('<h1>' + length + ' - ' + targetHTML + '</h1>');
        });
    });
</script>
```

8 이벤트 연결 범위 한정

◆ 이벤트 연결 범위 한정

- 어떠한 h1 태그를 선택해도 요소가 추가됨
- 이벤트 리스너에서 this 키워드가 #wrap 태그가 아니라 h1 태그라는 것을 주의

Header

1 - Header

2 - 1 - Header

3 - 2 - 1 - Header

4 - 1 - Header

8 이벤트 연결 범위 한정

◆ 이벤트 연결 범위 한정

- delegate 방식으로 연결한 on() 메서드의 이벤트 리스너 삭제

```
<script>
    $(document).ready(function () {
        // 이벤트를 연결합니다.
        $('#wrap').on('click', 'h1', function () {
            var length = $('h1').length;
            var targetHTML = $(this).html();
            $('#wrap').append('<h1>' + length + ' - ' + targetHTML + '</h1>');

            // 이벤트를 제거합니다.
            $('#wrap').off('click', 'h1');
        });
    });
</script>
```

9 마우스 이벤트

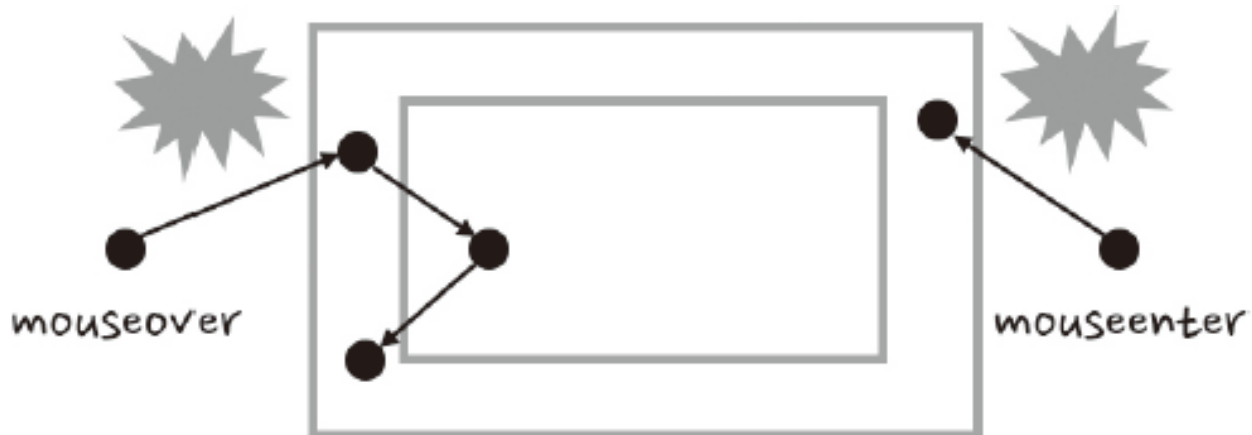
◆ 마우스 이벤트

이벤트 이름	설명
click	마우스를 클릭할 때 발생합니다.
dblclick	마우스를 더블 클릭할 때 발생합니다.
mousedown	마우스 버튼을 누를 때 발생합니다.
mouseup	마우스 버튼을 떼어낼 때 발생합니다.
mouseenter	마우스가 요소의 경계 외부에서 내부로 이동할 때 발생합니다.
mouseleave	마우스가 요소의 경계 내부에서 외부로 이동할 때 발생합니다.
mousemove	마우스를 움직일 때 발생합니다.
mouseout	마우스가 요소를 벗어날 때 발생합니다.
mouseover	마우스를 요소 안에 들어올 때 발생합니다.

9 마우스 이벤트

◆ 마우스 이벤트

- mouseover 이벤트와 mouseenter 이벤트의 차이
- mouseover 이벤트는 이벤트 버블링을 적용
→ 내부의 div 태그 안에 들어가도 이벤트를 발생
- mouseenter 이벤트는 문서 객체의 안에 있는지 외부에 있는지 따짐



10 키보드 이벤트

◆ 키보드 이벤트

이벤트 이름	설명
keydown	키보드를 누를 때 발생합니다.
keypress	글자가 입력될 때 발생합니다.
keyup	키보드를 떼를 때 발생합니다.

10 키보드 이벤트

◆ 키보드 이벤트

- textarea 태그에 keyup 이벤트를 연결
- keyup 이벤트가 발생하면 글자의 개수를 받아 출력

```
<script>
    $(document).ready(function (event) {
        $('textarea').keyup(function () {
            // 남은 글자 수를 구합니다. ex)
            var inputLength = $(this).val().length;
            var remain = 150 - inputLength;

            // 문서 객체에 입력합니다.
            $('h1').html(remain);
        });
    });
</script>
```

10 키보드 이벤트

◆ 키보드 이벤트

- keydown 이벤트 진행 순서

- 1 사용자가 키보드를 누릅니다.

- 2 keydown 이벤트가 발생합니다.

- 3 글자가 입력됩니다.

- 4 keypress 이벤트가 발생합니다.

- 5 사용자가 키보드에서 손을 뗍니다.

- 6 keyup 이벤트가 발생합니다.

- 입력한 글자 수를 표시해야 하므로 keyup 이벤트를 사용

11 윈도 이벤트

◆ 윈도 이벤트

- 윈도 이벤트는 윈도 객체만 사용할 수 있는 이벤트가 아니라 window 객체와 document 객체 이외에 img 태그 등이 사용할 수 있는 이벤트

이벤트	설명
ready	문서 객체가 준비 완료되면
load	윈도(문서 객체)를 불러들일 때 발생합니다.
unload	윈도(문서 객체)를 닫을 때 발생합니다.
resize	윈도의 크기를 변화시킬 때 발생합니다.
scroll	윈도를 스크롤할 때 발생합니다.
error	에러가 있을 때 발생합니다.

12 입력 양식 이벤트

◆ 입력 양식 이벤트

이벤트 이름	설명
change	입력 양식의 내용을 변경할 때 발생합니다.
focus	입력 양식에 초점을 맞추면 발생합니다.
focusin	입력 양식에 초점이 맞추어지기 바로 전에 발생합니다.
focusout	입력 양식에 초점이 사라지기 바로 전에 발생합니다.
blur	입력 양식에 초점이 사라지면 발생합니다.
select	입력 양식을 선택할 때 발생합니다(input[type="text"] 태그 및 textarea 태그 제외).
submit	submit 버튼을 누르면 발생합니다.
reset	reset 버튼을 누르면 발생합니다.

12 입력 양식 이벤트

◆ 입력 양식 이벤트


- submit 이벤트와 **기본 이벤트 제거** `return false;`
- submit 이벤트는 form 태그에서 발생하는 이벤트
- form 객체에 submit () 메서드를 연결
- 입력 양식의 유효성 검사를 할 때는 기본 이벤트를 제거해야 함

12 입력 양식 이벤트

◆ 입력 양식 이벤트

```
<script>
    $(document).ready(function () {
        $('#my-form').submit(function (event) {
            // 입력 양식의 value를 가져옵니다.
            var name = $('#name').val();
            var password = $('#password').val();

            // 출력합니다.
            alert(name + ' : ' + password);

            // 기본 이벤트를 제거합니다.
            event.preventDefault();  return false;
        });
    });
</script>
```

12 입력 양식 이벤트

◆ 입력 양식 이벤트

- check 속성 변경
- type 속성이 checkbox와 radio인 input 태그의 상태를 변경하는 이벤트는 click 이벤트가 아닌 change 이벤트

12 입력 양식 이벤트

◆ 입력 양식 이벤트

```
<script>
    $(document).ready(function () {
        $('#all-check').change(function () {
            if (this.checked) (true/false)
                $('#check-item').children().prop('checked', true);
            } else {
                $('#check-item').children().prop('checked', false);
            }
        });
    });
</script>
<body>
    <input type="checkbox" id="all-check" />
    <label>All</label>
    <div id="check-item">
        <input type="checkbox" />
        <label>A Option</label>
        <input type="checkbox" />
        <label>B Option</label>
        <input type="checkbox" />
        <label>C Option</label>
    </div>
</body>
```

제이쿼리 이미지 슬라이드 플러그인

- ◆ <https://tympanus.net/codrops/2014/06/26/draggable-dual-view-slideshow/>
- ◆ <http://idangero.us/swiper/demos/>
- ◆ <http://kenwheeler.github.io/slick/>
- ◆ <http://bqworks.com/slider-pro/>
- ◆ <https://tympanus.net/codrops/2014/03/13/tilted-content-slideshow/>